Coalition Game for Video Content Clustering in Content Delivery Networks

Nesrine Hassine, Pascale Minet, Mohammed-Amine Koulali, Mohammed Erradi, Dana Marinca, Dominique Barth

Coalition Game for Video Content Clustering in Content Delivery Networks

Nesrine Ben Hassine, Pascale Minet
Inria Paris, 2 rue Simone Iff, CS 42112,
75589 Paris Cedex 12, France
Email: nesrine.ben-hassine@inria.fr
pascale.minet@inria.fr

*Mohammed-Amine Koulali,
Mohammed Erradi
*Mohammed I University Oujda, Morocco
Mohammed V University, Rabat, Morocco
Email: m.koulali@ump.ac.ma, erradi@ensias.ma

Dana Marinca, Dominique Barth
University of Versailles Saint-Quentin,
78000 Versailles, France
Email: dana.marinca@uvsq.fr
Dominique.barth@uvsq.fr

***Abstract*—Game theory is a powerful tool that has recently been used in networks to improve the end users' quality of experience (e.g. decreased response time, higher delivery rate). In this paper, we propose to use game theory in the context of Content Delivery Networks (CDNs) to organize video contents into clusters having similar request profiles. The popularity of each content in the cluster can be determined from the popularity of the representative of the cluster and used to store the most popular contents close to end users. A group of experts and a decision-maker predict the popularity of the representative of the cluster. This considerably reduces the number of experts used. More precisely, we model the clustering problem as a hedonic coalition formation game where each coalition represents a cluster. The coalition game converges to a stable partition representing a solution of the problem considered. We compare the results of this approach with the clustering obtained by the K-means algorithm. We evaluate the impact of the content profile observation window considered to establish the clustering. We also evaluate the complexity of the proposed algorithm. Simulation results are obtained on traces of a real CDN. Finally, we extend the proposed approach to model an on-line clustering reflecting the CDN dynamics in terms of proposed contents and contents solicitations.**

Keywords*—*Content Delivery Network, clustering, YouTube, video content, coalition game.

I. Learning techniques in Content Delivery Networks

Content Delivery Networks, CDNs are becoming ever increasingly successful. For instance, YouTube, known to be the most popular for User-Generated Contents, serves 460 million visits per day, while Netflix, a Video-on-Demand system, serves 22 million visits per day. The price of this success lies in the large amount of traffic generated to allow end users to view video content when it pleases them. To reduce this traffic, caching techniques are used to store the video contents close to the end users. For an efficient management of the CDN and a high degree of end user satisfaction, video contents with the highest popularity must be cached.

In a previous paper [1], we showed how to use machine learning techniques to predict the popularity of video content in order to store the most popular content close to the users requesting it. For that purpose, several experts provide their popularity prediction to a forecaster that builds its own prediction based on the advice of its experts. However, due to the high amount of video contents in a CDN, it is not realistic to have a group of experts for each video content. In this paper, we present a solution to group together video contents showing some similarities in terms of request profiles. We use game theory as a framework to state our problem and find the most suitable clustering for the video contents considered. The simulations are based on real video profiles collected from YouTube.

This paper is organized as follows. In Section II, we present some examples where game theory is used in networks. In Section III, we model the clustering problem as a coalition game with as much players as the number of video contents to cluster. In Section IV, we compare the clustering obtained by this coalition approach with that obtained by the well-known K-means algorithm. We evaluate the impact of the size of the observation window and of the play order. The complexity of the coalition formation algorithm is computed to prove the scalability of this approach. Finally, we conclude in Section V.

II. Related work

Game theory is a powerful mathematical tool that studies the behavior of rational agents interacting in strategic scenarios to maximize their gains and minimize their costs. In cooperative games [2] the players form coalitions to enhance their collective and marginal benefits. In contrast, in non-cooperative games [3] players act selfishly and independently to enhance their own utility for a given profile of others' decisions. In strategic interactions, the gain of a particular player does not depend solely on this player's own decision. Indeed, the behavior of the other players involved will highly impact the outcome of the game (i.e. individual payoffs). Depending on the redistribution of the coalition gain, two sub-classes of cooperative games have been defined, namely: transferable and non transferable utility games. When the decision to join a given coalition is based only on which other players are already present, the cooperative game is said to be hedonic. If the value (gain) of a coalition is not super-additive the agents will rearrange into a set of disjoint collections according to a process called a partition formation game. The convergence of this game along with its stability are of great importance.

Game theory has recently been used in networks to model, analyze and provide solutions to various problems. The authors of [4] provide a game theoretical formulation of the proactive caching of videos on small base stations. The proposal aims to reduce the latency experienced by the users. Their solution in based on many-to-many matching games and the developed caching strategy satisfies up to three times more requests than random caching policy.

In [5] the authors proposed a coalition formation among secondary base stations (SBSs) in a cognitive radio network. The main objective of this work is to form coalitions between the SBSs to improve the accuracy of detecting primary users. The SBSs share their information through control channels with secondary users. The coalitions are formed based on the tradeoff between the gain from learning new channels and the cost of receiving inaccurate information.

A cooperative game model is proposed in [6] to investigate content production and sharing in P2P networks. An incentive-based mechanism is proposed to counteract the selfishness of individual peers. In this work, several incentive mechanisms such as cooperation, payments, repeated peer interaction, intervention, and enforced full sharing are compared.

The authors of [7] investigate coalition games for cooperative spectrum sensing. They propose maximizing the detection probability of the primary user presence while minimizing the probability of false alarms in single-channel cognitive networks. A distributed learning algorithm for Nash-stable coalition formation based on a sequence of switch rules is provided.

In [8], the authors introduce a coalition game model for self-organizing unmanned aerial vehicles collecting data from randomly located tasks in wireless networks. A hedonic game formulation is provided and the stability of the coalition formation game proved.

In this paper, we focus on hedonic coalition formation [8], where each player switches to an existing coalition that is preferred to the current one.

III. Coalition formation problem

A. Concepts and notations

We first introduce some notations before formulating the clustering problem as a coalitional game.

We consider n video contents. We denote i, $i \in [1, n]$ any video content in this set. We measure the **popularity** of each video content as its number of requests per day. Let $y_i(k)$ be the number of requests of video content i on day k.

We consider an **observation window** ow of size $size(ow)$. At day k, the observation window associated with a video content includes the days $k - size(ow)$, $k - size(ow) + 1$, ..., $k - 1$. We use the numbers of daily requests for all days in the observation window to perform content clustering. The idea consists in forming clusters of video contents with similar daily request profiles.

We define the **representative** of a set of video contents $rep(C)$, as the video content in C that minimizes the square of the distance of the other contents in C to itself:

$$\forall C, rep(C) = argmin_{i \in C} \sum_{j \in C, j \neq i} \sum_{k \in ow} (y_j(k) - y_i(k))^2.$$

The representative of any set C is dynamically selected among the members of C after each change in the membership of C. We also define the **maximum distance of a set C of video contents to its representative**. It is defined as follows:

$$Dmax(C) = \sum_{j \in C} \sum_{k \in ow} (y_j(k) - y_{rep(C)}(k))^2.$$

For each player i, we define $History(i)$ the history of i as the set of coalitions that player i left voluntarily.

A **round** is the time needed to allow each player to play exactly once. Hence, the number of rounds is the number of times a single player plays in the game.

B. Coalitional game problem and its solutions

Clustering of video contents is modeled as a coalitional game with n rational players, where n is the number of video contents to consider. The set of players is denoted $\{i, i = 1 \dots n\}$, where i is a video content.

At any time, the coalitional game ensures that any player belongs to exactly one coalition. Each coalition represents the set of video contents belonging to the same cluster.

Each player $i \in [1, n]$ defines a **preference relation** denoted $\geq_i$ over $\mathcal{C}_i$ the set of all possible coalitions to which player i can belong. Let any two coalitions C_1 and $C_2 \in \mathcal{C}_i$, $C_1 \geq_i C_2$ if and only if player i strictly prefers to belong to coalition C_1 rather than coalition C_2, or player i likes both coalitions equally.

A coalition formation game is said to be **hedonic** [8] if and only if the two following rules are met:
$Rule\ R1$: The preference of any player, among two coalitions it belongs to, depends only on the players present in the two coalitions considered.
$Rule\ R2$: The coalitions form as a result of the preferences of the players over their possible coalition set.

As a consequence, **a hedonic coalition formation game is defined by a set of players and the preference relation for each player**.

We assume that the payoff of any player $i \in [1, n]$, denoted $u_i(C)$ is equal to the payoff of the coalition C to which it belongs.We also assume that the preference of any player $i \in [1, n]$ for a given coalition depends on the payoff i that the coalition brings it. More precisely:

$$\forall (C_1, C_2) \in \mathcal{C}_i, C_2 >_i C_1 \ iff \ u_i(C_2) > u_i(C_1).$$

More precisely, the payoff $u_i(C)$ of each player $i \in C$ is defined according to Algorithm 1 as follows:
$u_i : \quad 2^n \rightarrow R$

Algorithm 1 Payoff of player i in coalition C

```
Compute Threshold of coalition C
if C ∈ History(i) then
   u_i(C) = −∞
else
   if size(C) = 1 then
      u_i(C) = −Threshold − 1
   else
      if Dmax(C) > Threshold then
         u_i(C) = −∞
      else
         u_i(C) = −Dmax(C)
      end if
   end if
end if
```

where $rep(C)$ is the representative of coalition C and $Dmax(C)$ denotes the maximum distance of contents in coalition C to the representative of this coalition. Algorithm 3 shows how the value of $Threshold$ is computed.

The representative of coalition C is selected according to Algorithm 2 as follows:

Algorithm 2 Selection of the representative of coalition C

if $size(C) = 2$ **then**
  if $(\sum_{j \in C, k \in ow} y_j^2(k)) \leq (\sum_{m \in C, k \in ow} y_m^2(k))$ **then**
    $rep(C) = m$
  else
    $rep(C) = j$
  end if
else
  $rep(C) = argmin_{i \in C} \sum_{j \in C, j \neq i} \sum_{k \in ow} (y_j(k) - y_i(k))^2$
end if

The value of $Threshold$ depends on the coalition C considered. It is computed according to Algorithm 3 as follows:

Algorithm 3 Computation of the Threshold value in coalition C

if $\sum_{k \in ow} y_{rep(C)}^2(k) < \alpha * size(ow)$ **then**
  $Threshold = \alpha * Drift^2 * size(ow)$
else
  if $\sum_{k \in ow} y_{rep(C)}^2(k) \geq \beta * size(ow)$ **then**
    $Threshold = Drift^2 * \beta * size(ow)$
  else
    $Threshold = Drift^2 * \sum_{k \in ow} y_{rep(C)}^2(k)$.
  end if
end if

We distinguish three cases for the computation of $Threshold$, depending on the sum of the square values of requests for the coalition representative in the observation window. If this sum is less than $\alpha * size(ow)$, this case occurs when a particular video content is no longer or very seldom requested, $Threshold$ is set to α times the size of the observation window ow multiplied by $Drift^2$, where $Drift$ is the maximum relative drift between the representative of C and a member of this coalition. We can take, for instance, $Drift = 10\%$. If on the other hand, the sum of the square values of the representative of C is very large, higher than $\beta * size(ow)$, then $Threshold$ is set to β times the size of the observation window ow times $Drift^2$. α and β are parameters determined empirically depending on the number of requests for contents considered meeting $\alpha < \beta$. When the sum of the squares belongs to the interval $[\alpha * size(ow), \beta * size(ow)]$, $Threshold$ gets a value proportional to the sum of the squares of the number of requests for the representative of coalition C. More precisely, $Threshold = Drift^2 * \sum_{k \in ow} y_{rep(C)}^2(k)$.

With this payoff function, each player is strongly discouraged from returning to a coalition belonging to its history. It is also discouraged from remaining alone in its coalition, as each player is encouraged to join a coalition where all the members are close to their cluster representative. This is captured by the payoff function (see the last instruction in Algorithm 1).

In a coalition game, a **partition** $\mathcal{P}$ is a set of disjoint coalitions such that any player $i \in [1, n]$ belongs to exactly one coalition in $\mathcal{P}$.

Each player plays one after the other, according to an ordered sequence. Let $\mathcal{P}$ denote the current partition formed by the coalitions that exist when any player i is playing. Player i in coalition C_1 wants to increase its payoff by joining another coalition C_2 that exists when it is playing but does not belong to its history: $C_2 \in \mathcal{P} \bigcup \{\emptyset\}$ and $C_2 \notin History(i)$, where $History(i)$ denotes the set of coalitions that i left.

The hedonic coalitional game is based on the switching rule. This rule corresponds to a selfish decision, where any player i decides to switch to another coalition in the current partition $\mathcal{P}$ independently of the effects on the other players, provided this switch increases the payoff of i:

Switching rule: Any player i leaves its current coalition C_1 to join coalition $C_2 \in \mathcal{P} \bigcup \{\emptyset\}$ and $C_2 \notin History(i)$ if and only if

$$u_i(C_2 \bigcup \{i\}) > u_i(C_1).$$

This can be also noted by:

$$C_2 \bigcup \{i\} >_i C_1.$$

Theorem 1. *Starting from any initial partition, this hedonic coalitional game always converges to a final partition.*

Proof: This hedonic coalitional game can be seen as an ordered sequence of partitions starting from the initial one, denoted $\mathcal{P}_0$. We then have $\mathcal{P}_0 \rightarrow \mathcal{P}_1 \rightarrow \mathcal{P}_2 \ldots \mathcal{P}_k \rightarrow \mathcal{P}_{k+1} \ldots$ where $\mathcal{P}_{k+1}$ is obtained from $\mathcal{P}_k$ after a player $i \in [1, n]$ has applied the switching rule. We observe that (1) the number of players is finite, (2) the number of possible coalitions each player i can belong to is finite: the size of $\mathcal{C}_i$ is finite, and (3) no player prefers going back to a coalition belonging to its history. It follows that the sequence of partitions obtained in the coalitional game is finite. Hence the game converges to a final partition. ■

Property 1. *The final partition obtained in this hedonic coalitional game is Nash-stable and individually stable [8]. In other words, no player can get a strictly higher payoff, even by joining another coalition.*

Proof: By contradiction, we assume there exists a player i that has an incentive to increase its payoff by leaving its coalition C_1 in the final partition $\mathcal{P}$ and joining the coalition $C_2 \in \mathcal{P} \bigcup \{\emptyset\}$ such that $C_2 \bigcup \{i\} >_i C_1$. This contradicts the fact that $\mathcal{P}$ is the final partition of the hedonic coalitional game. Hence, the final partition is Nash stable. ■

The final partition obtained in this coalition game is such that on the one hand, the number of coalitions in the final partition provides the number of clusters obtained, and on the other hand, each coalition belonging to the final partition represents the set of video contents belonging to the same cluster.

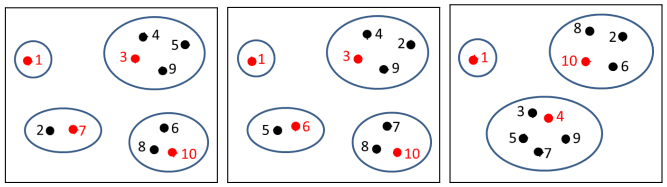
C. Discussion

Notice however that the final partition obtained depends on both the **initial partition** and the order in which the players play. In this coalition game where no clustering has been done before, it is natural to start from an initial partition where each video content is alone in its cluster. Different play orders may lead to different clusterings as illustrated in Figures 1a, 1b and 1c. Ten video contents are considered for clustering. Initially each of them belongs to a distinct cluster and the play order of the coalition game is Random, Rich-to-Poor and Poor-to-Rich, respectively.

The **Poor-to-Rich order** is obtained when players play according to the increasing order of their payoffs. The **Rich-to-Poor order** is obtained when they play according to the decreasing order of their payoff. More precisely and for both

orders, the play order in each round r is computed at the beginning of round r: the players are sorted according to their current payoff. Notice however that the **Random order** is computed only once at the beginning of the game and is used at each round of the game.

In Figures 1, we depict each coalition by a set with its representative represented in red while the other members are in black. We observe that the only common coalition to all three orders is the coalition $\{1\}$. The cluster centered around 3 exists in the Random and Rich-to-Poor orders, however its membership differs in both orders (i.e. see members 2 and 5). In this example, the Poor-to-Rich order leads to the smallest number of coalitions: three instead of four for the other orders. In the following, we will conduct more extensive simulations to compare these three orders (see Section IV-C).



a Random order b Rich-to-Poor order c Poor-to-Rich order

Fig. 1: Clustering of 10 contents.

The switching rule can be applied with different policies to join a coalition that improves the payoff of the player. We can distinguish:

- the **First-Coalition**, where the player joins the first coalition $C \in \mathcal{P}$ that improves its payoff. This policy is the simplest one and requires the least computation.
- the **Best-Coalition**, where the player computes its payoff if it joins each existing coalition and selects the coalition that provides the highest payoff.

The First-Coalition policy is preferred to the Best-Coalition policy, because of its better scalability. Scalability matters as the amount of video contents existing in a CDN is very high (e.g. > 5000).

IV. Performance evaluation

A. Simulation parameters

We first developed an extraction tool that extracts real traces of video contents stored in YouTube. We extracted 1000 video contents. The video contents used in the performance evaluation reported in this section were randomly chosen from those 1000 contents. We then developed a simulation tool for coalition games in Matlab. In all the simulations reported in this paper, each player applies the First-Coalition policy.

TABLE I: Simulation parameters

Observation window size	$size(ow) \in \{5, 7, 10\}$
Drift	10%, 30%
Parameters	$\alpha = 400$, $\beta = 1000000$
Number of video contents	$n \in \{20, 50, 100, 150, 200\}$
Choice of video contents	randomly extracted from YouTube
Policy	First-Coalition
Initial partition	each content in its own coalition
Play order	Random, Poor-to-Rich or Rich-to-Poor
Simulation result	average of 20 simulations

Each simulation is defined by its parameters such as the number and the identifier of the randomly selected video contents, the value of $Drift$, the size of the observation window ow, the play order. The values of these parameters are given in Table I. Each player $i \in [1, n]$ plays in sequence. The play order is either Random, Rich-to-Poor or Poor-to-Rich. It is fixed for the whole game.

Initially, each player is alone in its coalition. In other words, the initial partition in the coalitional game is given by $\{\{1\}, \{2\}, \ldots \{n\}\}$. There are as many clusters as video contents.

We are now able to study the impact of (1) the size of the observation window, (2) the play order, and (3) the number of video contents considered, on the clustering obtained. This clustering is qualitatively evaluated by:

- the number of clusters obtained,
- the average distance $Davg$ of the cluster members to their representative,
- the maximum distance $Dmax$ of cluster members to the representative of the cluster.

We evaluate the complexity of the coalition game in terms of number of rounds, number of switches and execution time. Finally, we compare the results obtained with those given by the well-known K-means algorithm.

B. Impact of the size of the observation window

The size of the observation window can range from the creation time of the video content up to the current time or it can be equal to the last week only, for instance. In this first series of simulations, we randomly select 50 video contents and vary the size of the observation window ow in the set $\{5, 7, 10\}$. Smaller values are not tested because they would have less practical interest since they would lead to a too frequent clustering. Simulation results reported in Table II show that the smallest size, 5, tends to increase the number and/or the size of the coalitions formed, while minimizing the average distance to the representative of the coalition. That is why in the following, we take a value of 5 for the size of the observation window.

TABLE II: Impact of the observation window size.

$size(ow)$	coalitions	Davg	Dmax
5	2.4	236.4	7173.4
7	2.3	1077	8252.3
10	1.3	731	1476.3

C. Impact of the play order and the number of video contents

In this second series of simulations, we evaluate the impact of both the number of video contents and the play order on the clustering in terms of number of clusters obtained, average and maximum distance to the representative of this cluster. The number of video contents ranges from 20 to 200. For each configuration defined by the number of video contents considered, we compare the results obtained for three play orders: Random, Rich-to-Poor and Poor-to-Rich. The average number of clusters obtained is depicted in Figure 2.

This number tends to increase with the number of video contents as long as this number is less than or equal to 150; There is no noticeable difference in the number of clusters obtained for 150 and 200 contents. This can be explained by the fact that there is a limited number of content profiles.

Contents are then grouped together in a number of clusters corresponding to this number of profiles. We observe that the play order has a very limited impact on the number of clusters obtained. For 20 and 100 contents, the number of clusters is identical for the three orders tested.

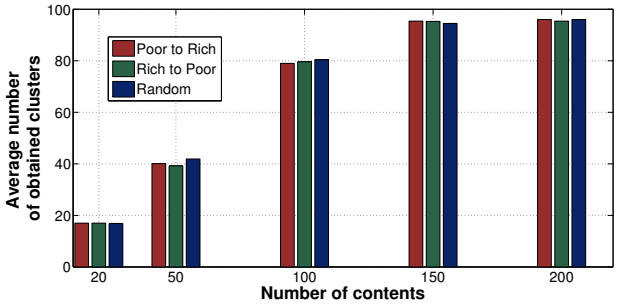


Fig. 2: Average number of clusters.

We also depict the average distance to the representative of the cluster in Figure 3, taking into account only clusters whose size is strictly higher than one. It tends to increase with the number of contents considered. The Rich-to-Poor order is the order minimizing the average distance obtained for all configurations tested. The reason is that rich players form coalitions with the smallest distance to their representative, subsequently poor players join these coalitions.

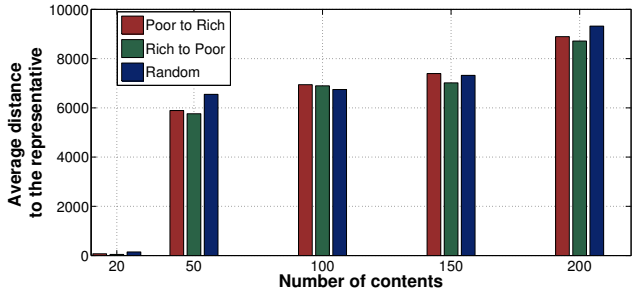


Fig. 3: Average distance to the representative of the cluster.

The maximum distance to the representative of the cluster is illustrated in Figure 4. Again, the Rich-to-Poor order outperforms the Random and Poor-to-rich orders in minimizing the maximum distance to the representative. This appears clearly for 200 contents. The reason is the same as for the average distance to the representative. As a conclusion regarding this series of simulations, we select the Rich-to-Poor order for further simulations.

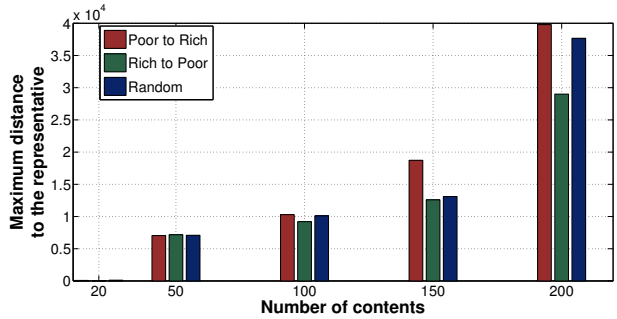


Fig. 4: Maximum distance to the representative of the cluster.

D. Complexity

It is interesting to know the complexity of this coalition game. We first evaluate the complexity by the number of rounds needed to get the stable partition as a function of the number of players (Figure 5). The three play orders lead to a very small number of rounds. Even for 200 contents, this number of rounds is less than 35 for the Rich-to-Poor and Poor-to-Rich orders. The Rich-to-Poor order, that is the order tending to minimize the maximum distance to the representative, gives very good performances.

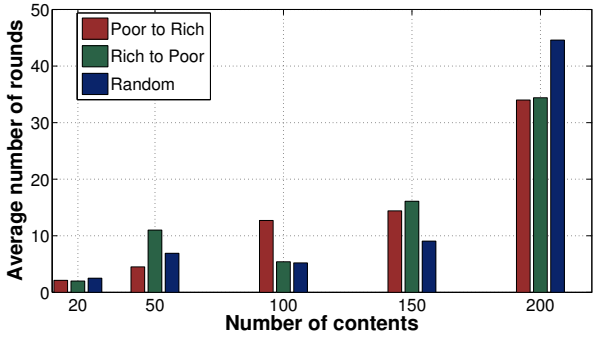


Fig. 5: Average number of rounds.

Another interesting parameter to evaluate the complexity of the coalition game is the total number of switches done by the players.

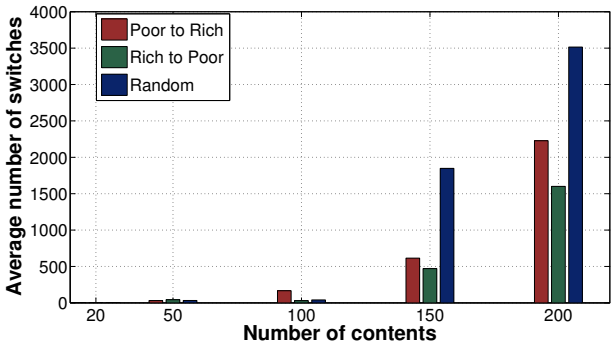


Fig. 6: Average number of switches.

This is the number of times the switching rule is applied in the game. Knowing the number of rounds R, the total number of switches, denoted S, is upper bounded by $n*(R-1)$, where n is the number of players. The term -1 is introduced to take into account the fact that during the last round no player is able to increase its current payoff by joining an existing coalition. Simulation results corroborate this bound. Here again, we observe the very good performances for the Rich-to-Poor order that minimizes the total number of switches (Figure 6): less than 1520 for 200 contents. This is explained by the fact that since rich players, belonging to a coalition where the maximum distance is small, select the existing coalitions that increase their payoff, poor players are encouraged to join these coalitions and it offers them fewer possibilities in the future to improve again their payoff. This is reflected by the difference $S - n * (R - 1)$ that accounts for the number of times a player is unable to increase its current payoff. For instance, with 200 contents and the Rich-to-Poor order, the difference is $34 * 200 - 1520 = 5280$.

For practical reasons, it may be necessary to know the time needed to obtain the final result. That is why we also measure the time spent on computing the final result in Figure 7. Up-to-now the Rich-to-Poor order outperforms the other play orders. However, this could be questioned if it was achieved at the cost of a large computation time because of the computation of the play order at each round. Simulation results show that this is not the case. Even for 200 contents, the clustering is obtained in less than 1000 seconds (i.e. 16 minutes) on a laptop, processor Intel with 8-Core, 2.7 GHz and 8 Gb of memory. Hence, this coalition game is scalable.

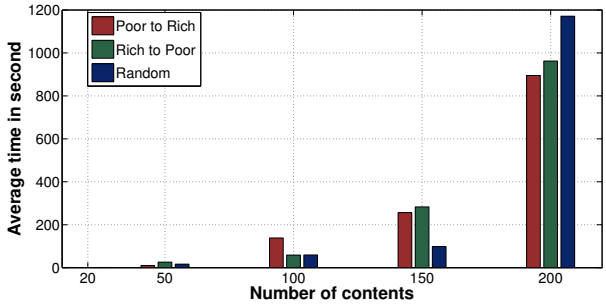


Fig. 7: Average time in seconds.

E. Comparison with the K-means clustering algorithm

Before comparing the simulation results obtained with those obtained by the well-known K-means clustering algorithm [9], we present the principles of the K-means algorithm.

1) Presentation of the K-means clustering algorithm: This algorithm is widely used for cluster analysis in data mining. K-means clustering partitions observations into K clusters such that each observation belongs to the cluster with the nearest mean. As a result, the data space is partitioned into K Voronoi cells.

The K-means algorithm proceeds by iterations. Each iteration t consists of two steps:

- *Assignment step*: Assign each observation x_p to the cluster $C_i^{(t)}$, with $1 \leq i \leq K$, whose mean $m_i^{(t)}$ at iteration t yields the least within-cluster sum of squares (WCSS). Since the sum of squares is the squared Euclidean distance, this is intuitively the "nearest" mean.

$$C_i^{(t)} = \left\{ x_p : \left\| x_p - m_i^{(t)} \right\|^2 \leq \left\| x_p - m_j^{(t)} \right\|^2 \forall j, 1 \leq j \leq K \right\},$$

  where each x_p is assigned to exactly one cluster $C_i^{(t)}$.
- *Update step*: Compute the new means $m_i^{(t+1)}$ as the centroids of the observations in the new clusters.

$$m_i^{(t+1)} = \frac{1}{|C_i^{(t)}|} \sum_{x_j \in C_i^{(t)}} x_j.$$

Since both steps optimize the WCSS objective, and there only exists a finite number of partitions, it has been proved in [9] that this algorithm converges to an optimum, but without any guarantee of finding the global optimum.

Initially, K observations are randomly selected from the data set and are used as centroids. We show with the example depicted in Figure 8 that the clustering obtained with the K-means algorithm depends on the random selection of K observations chosen as initial centroids. We consider 10 video contents and set the value of K to 4. The left part and the right part of Figure 8 illustrate two different results for two initial selections of centroids given by $\{2, 10, 8, 9\}$ and $\{2, 10, 4, 5\}$). These results are obtained by two successive executions of K-means on the same set of video contents.

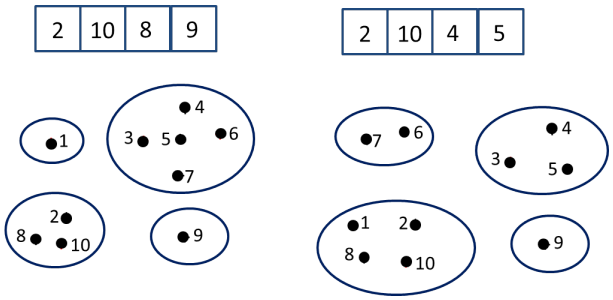


Fig. 8: Impact of the initial random selection in K-means.

In this algorithm, K, the number of clusters built by the K-means algorithm, is given as an input of the algorithm. The main difficulty is how to fix K a priori, without having any idea of the result of the clustering. An inadequate value of K may lead to a clustering of poor quality, as depicted in Figure 9, where we consider the same set of contents as in Figure 8, but we give a value of $K = 2$, with the selection of$\{4, 5\}$ as initial centroids. We get two coalitions with a maximum distance to the representative equal to 35737.

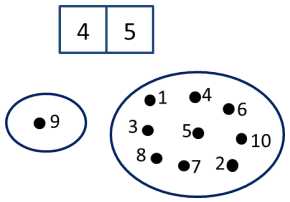


Fig. 9: Impact of the value of K on the K-means clustering.

We notice that all these drawbacks of the K-means clustering algorithm do not exist with the hedonic coalition formation game.

2) Comparative results with K-means clustering: To have a significant comparison of the clusterings obtained by the coalition game and K-means, we first run the coalition game and obtain a certain number of coalitions n_C. We then run the K-means algorithm with $K = n_C$. We consider a number of video contents in the set $\{20, 50, 100\}$. Simulation results are reported in Table III and the associated execution times are given in Table IV.

TABLE III: Comparison between Game and K-means.

	Cluster		Davg		Dmax	
nb. contents	Kmeans	Game	Kmeans	Game	Kmeans	Game
20	7.4	17	9.21E+04	2.64	8.01E+05	8.8
50	17	41	4.56E+07	4.11E+03	9.63E+09	1.02E+05
100	28.6	79	6.20E+07	1.9E+03	1.28E+09	2.4E+03

For both K-means and the coalition game, the average number of clusters increases with the number of contents in $\{20, 50, 100\}$. We observe that, on the one hand, the number of clusters given by the coalition game is higher than with K-means and, on the other hand, the maximum distance to the

representative is much smaller: 91 compared with $1.28E+09$ for 100 contents. This means that the clustering performed by the coalition game is much more accurate. With regard to the time needed to obtain the results, K-means needs more time when the number of contents is higher than or equal to 50. For instance, it is 3.5 times the time needed by the coalition game for 50 contents.

TABLE IV: Execution times for Game and K-means.

	Time(s)	
nb. contents	Kmeans	Game
20	3.47E-03	1.04
50	44.66	38.10
100	1.44E+02	4.11E+01

As a conclusion, since the coalition game is able to quickly provide a clustering, we recommend using it to get the number of clusters n_C and then to run K-means with $K = n_C$.

F. Price of Anarchy

On the one hand, the coalition game is played as a distributed multi-agent system, where each agent is a player acting on the behalf of a video content. On the other hand, K-means is a centralized clustering algorithm that is widespread in data mining. It is therefore logical to evaluate the Price of Anarchy, denoted PoA, as the ratio between the solution provided by the coalition game and that provided by K-means. To take into account the three comparison criteria given previously, we define PoA as a 3-dimensional vector:

- The first component of this vector gives the ratio of the average distance within a cluster.
- The second component gives the ratio of the maximum distance between two contents belonging to the same cluster.
- The third component gives the ratio of execution time needed to obtain the clustering.

In the three components, the ratio is obtained when considering the distributed coalition game as the numerator and the centralized K-means algorithm as the denominator. Table V gives the Price of Anarchy for 20, 50 and 100 video contents.

TABLE V: Price of Anarchy.

	Davg	Dmax	Time
20	2E-05	10.9E-04	300
50	9E-05	10^{-03}	0.85
100	3E-05	187.5E-04	0.285

We observe that with regard to both the average distance and the maximum distance to the representative of the cluster, the Price of Anarchy is excellent for the coalition game. Furthermore, for a number of contents higher than or equal to 50, the Price of Anarchy is clearly in favor of the coalition game.

G. Advantages of the coalition game

Modeling the clustering problem as a coalition game has many advantages. Three of which we give below:

- It represents an elegant way to model the problem under consideration: constraints are naturally taken into account: a video content belongs to exactly one cluster and each cluster contains at least one video content.
- The approach proposed is scalable with regard to the number of players. This approach does not require the enumeration of all possible coalitions (i.e. 2^n for n players). A player belonging to coalition C_1 has only to pick a coalition C_2 in the current partition $\mathcal{P}$ such that the switching rule is met.
- The approach is simple: each player runs a very simple algorithm. Each player applies only one rule: the switching rule.

V. Conclusion

Clustering is frequently used in data mining. In this paper, we focused on clustering in content delivery networks in order to predict the popularity of video contents and improve cache management. The original contribution of this paper is to model clustering as a coalition formation game where the players are the video contents. We proved that this game always converges to a stable partition consisting of different clusters. We determined the best size of the observation window and showed that the play order minimizing the maximum distance to the representative of the cluster is the Rich-to-Poor order, whatever the number of video contents in the interval $[20, 200]$. The complexity of the coalition game remains very light. Convergence is obtained in a small number of rounds (i.e. less than 35 rounds for 200 video contents). A comparison with the K-means algorithm allowed us to determine the Price of Anarchy. This price is clearly in favor of the coalition game for the average and maximum distances to the representative of the cluster. From the execution time point of view, it is also in favor of the coalition game when the number of contents is higher than or equal to 50. Furthermore, the coalition game can be used to quickly determine the best value of K that is required as an input parameter of the K-means algorithm. Simulation results show that the coalition game is scalable and provides very good performances.

References

[1] N. Ben Hassine, D. Marinca, P. Minet, D. Barth "Popularity Prediction in Content Delivery Networks", PIMRC 2015, Hong Kong, August 2015.

[2] M. J. Osborne, A. Rubinstein. A course in game theory. MIT press, 1994.

[3] T. Basar, G. J. Olsder. Dynamic noncooperative game theory. Vol. 200. London: Academic press, 1995.

[4] K. Hamidouche, W. Saad, M. Debbah. Many-to-many matching games for proactive social-caching in wireless small cell networks. In 12th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), pp. 569-574, 2014.

[5] W. Saad, H. Zu, T. Basar, A. Hjrungnes, J. Bin Song, *Hedonic Coalition Formation Games for Secondary Base Station Cooperation in Cognitive Radio Networks*, WCNC 2010, Sydney, Australia, April 2010.

[6] J. Park, M. Van der Schaar. A game theoretic analysis of incentives in content production and sharing over peer-to-peer networks. IEEE Journal of Selected Topics in Signal Processing, Volume 4, Issue 4, 704-717, 2010.

[7] W. Saad, H. Zu, T. Basar, M. Debbah, A. Hjrungnesng, *Coalition formation games for collaborative spectrum sensing*, IEEE Trans. on Vehicular Technology, vol. 60, no. 1, 2011.

[8] W. Saad, H. Zu, T. Basar, M. Debbah, A. Hjrungnesng, *Hedonic coalition formation for distributed task allocation among wireless agents*, IEEE Trans. on Mobile Computing, vol. 10, no. 9, Sep. 2011.

[9] J. B. MacQueen, *Some Methods for classification and Analysis of Multivariate Observations*, Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, 1:281-297, 1967.

[10] W. Saad, H. Zu, M. Debbah, A. Hjrungnes, T. Basar, *Coalition game theory for communication networks: A tutorial*, IEEE Signal Processing Mag., Special issue on Game Theory in Signal Processing and Communications, vol. 26, no. 5, pp. 7797, Sep. 2009.